6. Memoria sistemelor de calcul Memoria interna a calculatoarelor

6.1. Reprezentarea datelor în memorie, parametrii ce caracterizeaza memoria.

Datele, cât si instructiunile, constituite din litere, cifre si caractere speciale sunt recunoscute de sistemul de calcul numai sub forma unor succesiuni de cifre binare, numite biti (bit sau binary digit).

Unitati de masura folosite

O succesiune de 8 biti se numeste byte sau octet, fiind cea mai mica unitate de date ce poate fi reprezentata si adresata de catre memoria unui sistem de calcul.

Multiplii byte-ului sunt:

1 Kilobyte = 1024 bytes (2^{10} bytes)
1 Megabyte = 1024 KB (2^{20} bytes)
1 Gigabyte = 1024 MB (2^{30} bytes)
1 Terrabyte = 1024 GB (2^{40} bytes)
1 Petabyte = 1024 TB (2^{50} bytes)
1 Exabyte = 1024 PB (2^{60} bytes)
1 Zettabyte = 1024 EB (2^{70} bytes)
1 Yottabyte = 1024 ZB (2^{80} bytes)

Reprezentarea în memorie a datelor se realizeaza la nivel de:
- byte (octet)
- cuvânt de memorie – 2bytes = 16 biti,
- dublu cuvânt – 4 bytes = 32 biti,
- cuvânt quadruplu – 8 bytes = 64 biti.

Extinderile multimedia (MMX – Multimedia Extensions) utilizate de microprocesoarele Intel si AMD au introdus suplimentar 4 tipuri de date ce se prelucreaza în registru de 64 biti:
- byte împachetat ce prelucreaza în grupuri de câte 8,
- cuvânt împachetat prelucrat în grupuri de 4,
- cuvânt a carui prelucrare se face în grupuri de 2,
- cuvânt quadruplu.

Parametrii ce caracterizeaza memoria

Memoria sistemului de calcul este caracterizata de urmatorii parametrii mai semnificativi:

Capacitatea - reprezinta numarul de bytes pe care îi poate stoca memoria la un moment dat. Acest numar este foarte mare si se exprima de regula prin multipli de bytes: K, M, G, T.

a) Timpul de acces – reprezinta intervalul dintre solicitarea unor date / informatii din memorie si obtinerea acesteia: $\Delta t = t_2 - t_1$ t_2 – momentul solicitarii ; t_1 – momentul obtinerii.

b) Modelul de organizare si adresare. Din punct de vedere al modului de organizare, memoria sistemelor de calcul include 2 niveluri:

Memoria interna Memoria interna – nimita si memorie principala, ce pastreaza datele si informatiile pe parcursul executiei de catre CPU. O data terminata executia datelor vor fi extrase din memoria principala (privita ca o memorie de lucru) si transferate catre dispozitivele de afisare a rezultatelor sau catre memoria externa. Aceasta are o capacitate mica si un timp de acces redus.

Memoria externa e numita si memoria auxiliara pastreaza datele/informatiile si programele pe suporturi externe, în vederea unor prelucrari ulterioare sau arhivarii; capacitatea de memorie este mare, practic nelimitata, în functie de suporturile disponibile, iar timpul de acces este mult mai mare decât la memoria interna.

În cazul memoriei virtuale, memoria externa se poate constitui ca o extensie a memoriei interne.

Din punct de vedere al accesului, memoriile sunt:
- cu acces direct, aleator, la care timpul de acces la orice adresa din memorie este acelasi;
- cu acces pozitional, la care timpul de acces la o anumita adresa este determinat de o serie de deplasari relative fata de adresa de la început.

a) Tehnicile de introducere/extragere a datelor/informatiilor, sau pot fi seriale sau paralele,

b) Viteza – reprezinta numarul de bytes ce se transfera în/din memorie într-o unitate de timp (rata de transfer).

6.2. Memoria interna

Memoria interna este componenta sistemului de calcul care stocheaza instructiunile de prelucrat, rezultatele intermediare si finale ale programelor în executie la un moment dat. Datele programului în executie trebuie sa se stocheze în memoria interna, care este o memorie de acces direct, pentru a furniza unitati de prelucrare, într-un timp minim, datele solicitate; astfel timpul de acces al memoriei interne trebuie sa fie mai mic sau cel mult egal cu timpul necesar unitatii centrale de prelucrare pentru a executa o instructiune ($t_a \leq t_{CPU}$).

Schimbul de date / informatii cu memoria este redat în fig. 6.1.

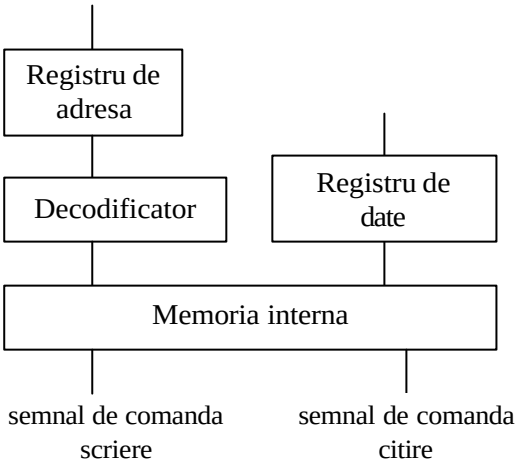

Fig. 6.1 Scrierea si citirea în / din memoria interna

Localizarea unei date în memorie se realizeaza prin specificarea adresei associate introduce într-un registru de n biti, care va putea identifica prin intermediul decodificatorului 2^n locatii de memorie.

Intervalul de timp necesar unei referiri la memorie se numeste ciclu de memorie (a nu se confunda cu timpul de acces), pe parcursul caruia continutul registrului de adresa ramâne nemodificat.

Dispozitivele fizice care alcatuiesc memoria interna trebuie sa îndeplineasca niste cerinte: existenta de 2 stari stabile, pentru a putea memora date, volum mic, cost cât mai scazut, timp de acces redus, realizarea modulara cu posibilitati de extindere.

Aceste deziderate au condus, în sistemele de calcul, la realizarea de memorii cu inele de ferita sau filtru magnetic, în momentul de fata fiind realizate cu circuite semiconductoare integrate, ce asigura un timp de acces de ordinal nanosecundelor. În esenta, memoriile constau din reunirea unor circuite basculante pe o singura pastila de siliciu, ce poate asigura o densitate a componentelor mare pe unitatea de volum. Calculele binare sunt asezate în grupuri pe linii si coloane, asa cum se observa în fig. 6.2.

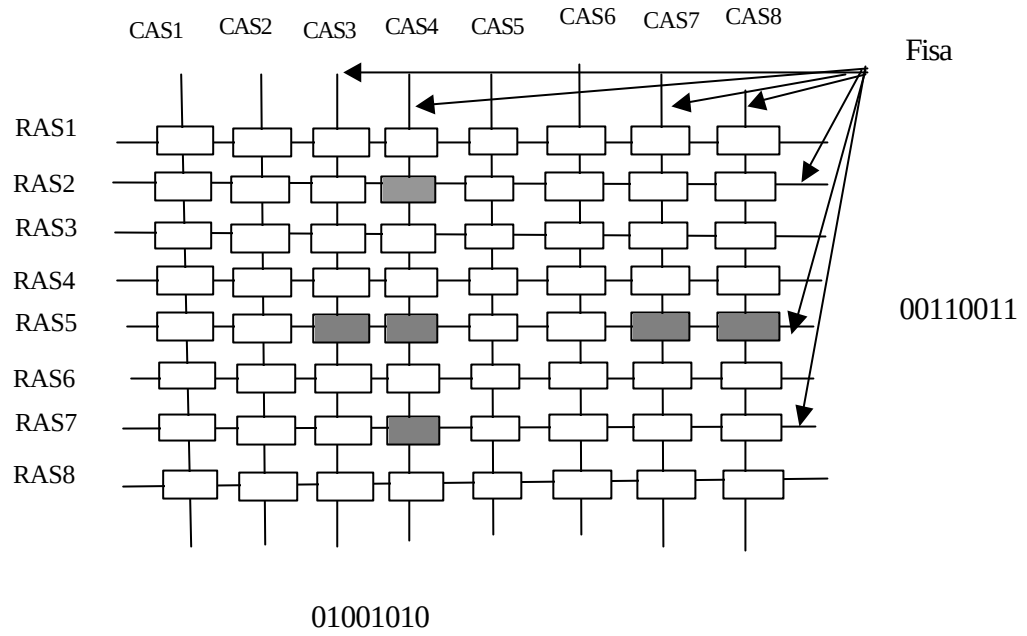

Fig. 6.2. Organizarea memoriei integrate

Încarcarea celulelor binare se realizeaza trimitând curent prin liniile si coloanele de selectare. În punctele în care firele încarcate electric se intercaleaza celulele binare sunt pozitionate pe "1", celelalte ramânând pe "0".

Memorii RAM

Se considera *"celula de memorie"* circuitul elementar care realizeaza memorarea unui bit. Acesta se poate realiza cu tranzistoare bipolare sau cu tranzistoare MOS. Circuitul de memorie propriu-zis este aranjat sub forma unei matricii în ale carei noduri gasesc celulele de memorie. Structura circuitului de memorie având dimensiunile n x m este data în fig. 6.3.

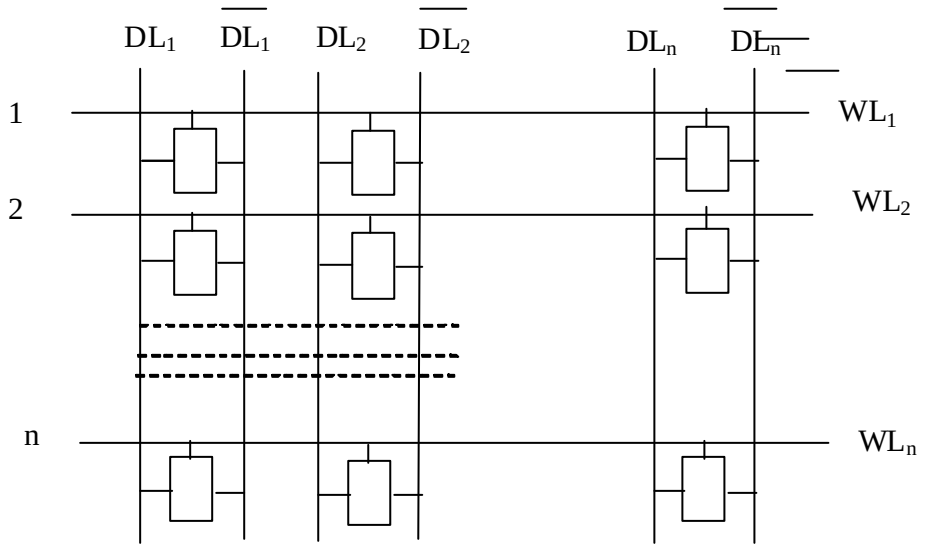

Fig. 6.3. Structura unei matricii de memorie

Pentru accesul la celula de memorie (i, j) se face selectia liniei WL_i si selectia coloanei DL_j, $\overline{DL_j}$. Capacitatea memoriei este data de produsul m x r.

Exista 2 categorii de memorii. RAM – memorii RAM cu acces dinamic (DRAM – Dynamic Random Acces Memory) al carei continut dispare daca prin semnalele de comanda nu se specifica încarcarea celulelor cu un anumit continut.

Pentru a-si îndeplini functia, fie chiar si temporar, sunt necesare comenzi care solicita conservarea continutului anterior, comenzi care solicita operatie numita reîmprospatare (refreshing memory). Reîmprospatarea este deci operatia ce consta în citirea periodica a datelor din memorie si reînscrierea lor la aceleasi adrese. Operatia se realizeaza automat la intervale de milisecunde. De exemplu, la SIMM RAM de 8M necesita reânprospatarea la fiecare 32ms.

Memorii SRAM (StatiC Random Acces Memory) – memorii cu acces static; pastreaza continutul celulelor binare, având o functionare asincrona.

Circuite cu memorii RAM

În fig. 6.4 sunt evidentiate blocurile componente necesare functionarii unui circuit de memorie RAM. Matricea de memorie contine la intersectia unei linii cu o coloana o celula de un bit. Decodificatoarele activeaza liniile de la selectie cuvânt $WL_0 – WL_p$ (unde $p = 2^n$) si pe de alta parte circuitele de citire pentru fiecare coloana, pe liniile $DL_0 – DL_q$ (unde $q = 2^m$).
Semnalele sunt transmise în exterior prin intermediul unor circuite de I/E. circuitul de control activeaza ansamblul de circuite interne ce poate participa la realizarea functiei (citire sau scriere). În memoriile RAM dinamice circuitul de control este mai complicat deoarece genereaza secventa de citire a informatiei din celula si revizuirea acesteia. Pentru stocarea simultana a mai multor biti se multiplica cu numarul de biti toate circuitele în afara de decodificatoare si circuite de control.

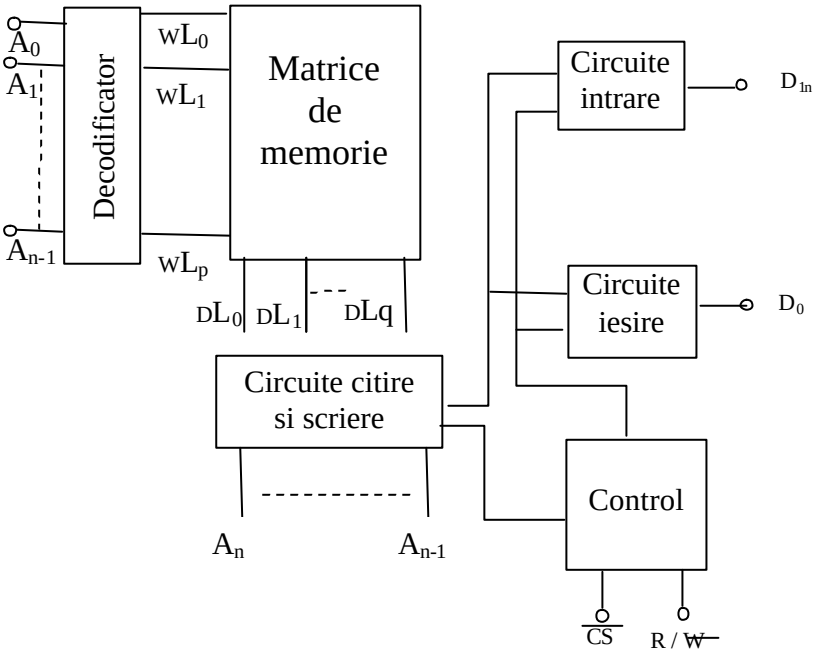

Fig. 6.4 Schema bloc a unui circuit de memorie de 1 bit.

Matricea de memorie are o structura interna ce se refera în geometria externa A memorie prin numarul de biti din cuvânt. Împartirea pe linii si coloane a acestei matricii nu este sesizabila din exterior, adresele $A_0 – A_{n-1}$ fiind tratate nedistinctiv în afara.

6.3. Memoria RAM. Formate fizice si logice.

6.3.1. Cip uri de RAM. Detectarea si corectarea erorilor.

O data cu aparitia microprocesoarelor pe 32 de biti, necesarul de memorie a devenit tot mai mare, iar cipurile individuale au devenit neconvenabile si impracticabile. Un bank tipic era alcatuit din 36 de cipuri, astfel ca mai multe bankuri ar fi acoperit tot spatiul de pe o placa. Alternativa a constituit-o modulul de memorie pe o singura placuta de circuit. Modulele de memorie sunt prevazute cu conectori CELP (Card Edge Low Profile) având 30-83 de contacte pe fiecare parte a modulului. SIMM-ul leaga unele contacte de pe ambele parti, iar DIMM-ul pria semnalele separate de pe fiecare parte a placii. SIPP-urile sunt cel mai adesea lipite pe placa de baza.

Cele mai frecvente tipuri de module sunt SIMM-uri cu 30 si 72 de pini. Smoll Outline DIMM (SODIMM) cu 72 de pinii si DIMM–urile cu 168 de pini.

Cele mai multe PC-uri verifica fiecare lot din memoria RAM de fiecare data când se porneste sau când se reactiveaza calculatorul, dar pot trece peste aceasta verificare initiala.

Controlerul de memorie face totalul celor 9 biti memorati în fiecare byte (incluzând si bitul de paritate) verificând ca acesta sa fie impar. În caz contrar invalideaza datele memorate.

Unii producatori de cipuri RAM au abandonat verificarea paritatii deoarece creste costurile cu 10–15%, în afara de aceasta adaugându-se si posibilitatea unei compactari a cipurilor utila pentru notebook-uri .

Exista si PC-uri care au implementata paritatea falsa (Fake Parity) care emite un semnal ce atesta paritatea fara sa realizeze o verificare efectiva.

Datorita faptului ca cipurile care genereaza paritate falsa sunt ieftine, ele sunt utile în PC-urile care au implementat mecanismul de detectare a paritatii. În general, modulele cu paritate falsa apar identice cu cele cu modulele cu paritate. Singurul mod de a le testa este cu un tester de SIMM.

Exista si metode care pot efectua atât detectarea cât si corectarea unor erori. Un exemplu îl constituie procedeul ECC (Error Correction Code) ce necesita 3 biti suplimentari pentru un byte memorat. Procedeul poate localiza bitul care este eronat. IBM folosea ECC pe computerele proiectate pentru a fi servere.

Pentru a obtine un timp de acces mic, este necesara realizarea unor cipuri pentru care se va urmari o micsorare a întârzâerilor interne. Cele mai rapide DRAM-uri au un timp de acces de 50-45µs. Cheia problemei consta în modul de aranjare a bitilor si a accesului.

Pentru a mentine un numar scazut de legaturi, de obicei liniile de adresare sunt multiplexate. Astfel acelasi set de linii serveste si pentru adrese de linii si de coloane.

SIMM – Single-in-Line Memory Module.

Primul SIMM cu 30 de pinii permitea o stocare de 9 biti pentru fiecare byte, folosind un bit extern pentru verificarea paritatii.

Înbunatatirea tehnologiei memoriilor a condus la cipuri fara verificarea paritatii, SIMM-ul cu 8 biti devenind uzual.

Un sistem fara paritate ignora semnalele de paritate de la un SIMM cu 9 biti (pinii 26, 28, 29) dar un SIMM cu 8 biti nu poate substui un SIMM cu 9 biti pentru ca un sistem cu verificarea paritatii va detecta eroare de paritate (absenta bitului suplimentar).

Unele sisteme permit întreruperea verificarii paritatii, iar alte întrerup verificarea automat când detecteaza un SIMM fara paritate.

Cea mai mare capacitate întâlnita la un SIMM cu 30 de pini a fost 4 MB/modul.

Pentru magistrale mari de date a fost proiectat SIMM-ul cu 72 de pini care încorpora 4 bankuri de memorie în acelasi modul. 4 din pinii SIMM-ului cu 72 de pini sunt folositi pentru a indica viteza de transfer a modului de memorie (timpi de acces 50, 60, 70, si 80 µs.

SIMM-urile pot fi cu paritate si fara paritate; cele cu paritate sunt circuite cu 36 de biti iar cele fara paritate cu 32 de biti.

#DIMM – Dual–in-Line Memory Module

O data cu aparitia procesoarelor Pentium si Pentium PRO chiar si SIMM-urile cu 72 de pini si-au pierdut din utilitate. Pentru magistrale de date de 64 de biti, acestea necesitau 2 SIMM-uri cu 72 de pini, la un bank de memorie asa cum 486 necesita 4 SIMM-uri cu 30 de pini. De aceea s-au realizat module cu 168

de pini pozitionati pe ambele parti ale modulului de memorie. Datorita numarului mare de pini, DIMM-urile au o lungime de 5,25 inch si înaltimea de 1 inch. Pinii 79, 80, 81, 82, 163, 164, 165, 166 indica intervale de transfer disponibile (de fapt PD6 si PD7(165,82) sunt importante, ceilalti pinii fiind identici.). Aceste viteze sunt de 60, 70, 80 µs.

#SODIMM – Smoll Outline DIMMs

SIMM-urile cu 72 de pini sunt prea mari pentru laptop-uri. Printr-o modificare de conector s-a redus lungimea modului la ½ (2,35 inch), fiind echivalent cu un SIMM cu 72 de corectori.

#SIPP – Single Line Pine Packag Module

Electric ele sunt identice cu SIMM-urile cu 30 de pini, dar fizic sunt mai mari. Desi din punct de vedere functional, SIMM si SIPP –urile pot avea aceleasi cipuri de memorie tehnologic si capacitate, între ele exista diferente ce le fac incompatibile.

6.3.2. Formate logice FPM, EDO, SDRAM, EDRAM, CDRAM, RAMBUS, VRAM, WRAM

Încarcarea cipurilor de memorie prin adunarea liniilor si coloanelor consuma timp de ordinul monosecundelor. Daca se adauga si timpul necesar reînprospatarii se obtin limitele performantei cipului de memorie.

Producatorii au dezvoltat tehnologii ca sa depaseasca aceste limite orientându-se asupra modului în care sunt procesate datele intern.

Static Columm RAM

Primele memorii au folosit tehnologia SCRAM, ce efectua citirea unei coloane de memorii si scrierea adresei pe linia de adresa a cipului, trimitând apoi semnalul CAS. O data ce coloana a fost înregistrata se poate trimite o noua adresa ce va indica o linie, activând RAS. În tot acest timp semnalul CAS este mentinut activ.

FPM (Fast Page Mode)

Tehnologiile FPM folosesc o varianta a acestei strategii. Controlerul de memorie trimite mai întâi o linie de adresa apoi activeaza RAS. Cât timp RAS e activ, se trimite o adresa a semnalului CAS pentru a indica o anumita celula. Daca RAS este tinut activ, controlerul poate trimite una s-au mai multe adrese, urmate de un impuls al semnalului CAS, pentru a indica celule din cadrul aceleasi linii.

În tehnologia de adresare a memoriei linia este numita pagina, iar tipurile de cipuri ce permit aceste operatii sunt numite page-mode–RAM. PC-ul poate astfel accesa mai rapid celulele dintr-o pagina cu un timp de acces de 25-30 µs.

EDO (Extended Data Out)

Este înca o tehnologie destul de rapida pentru calculatoarele actuale. Memoria EDO lucreaza cel mai bine în combinatie cu o memorie cache. În esenta EDO este o varianta a memoriei fAst-page mode (care permite accesul separat la bitii unei pagini fara a genera întârzieri). Memoria EDO modifica cuanta de timp alocata pentru semnalul CAS. Linia de date mai ramâne activa un timp scurt dupa ce linia CAS e dezactivata. Se elimina astfel timpul de asteptare necesar unui ciclu separat de citire/scriere, deci se pot citi/scrie date la viteza cu care cipul poate sa selecteze adresele. Cele mai multe cipuri au un delay de 10µs necesar între emiterea adreselor pe coloana.

Burst EDO DRAM

Pentru a câstiga viteza cu EDO, MicroTechnology a adaugat circuite cipului pentru al face compatibil cu modul burst folosit de µP Intel începând cu 486. BEDO realizeaza operatii de citire si scriere în cicluri de câte 4 cifre, numite burst. BEDO este relative usor de fabricat pentru ca necesita un numar de

schimbari minime fata de EDO. Cipul de siliciu contine un fuzibil care determina daca cipul va functiona ca EDO sau ca BEDO. El îsi poate schimba starea ireversibil prin arderea fuzibilului.

SDRAM (Synchronous RAM)

Datorita multiplexarii, circuitele de memorie nu pot oferi date sincron cu µP. Adunarea normala cere cicluri alternante, dar prin reproiectarea interfetei de baza cipurile din memorie pot accesa date la fiecare ciclu de ceas. Aceste memorii se numesc DRAM sincron. Ca si procesoarele superscalare, cipurile SDRAM sunt realizate cu stadii de operare multiple si independente, astfel încât cipul sa poata accesa a doua adresa înainte sa se încheie procesarea primeia. SDRAM are viteze de transfer de 10µs, ceea ce le permite sa utilizeze magistrale de memorie de 100 MHz.

SDRAM-urile nu vor opera la frecvente mai mari de 100- 133 MHz pentru ca slot-urile SIMM-urilor devin nesigure la frecvente mai mari.

EDRAM (Enhanced RAM)

EDRAM-urile sunt mai rapide si se obtin din DRAM-urile obtinute prin adaugarea unor blocuri mici de memorie cache statica pe cip. Cache-ul opereaza la viteza mare – usual 1µs – astefel încât pot sa acopere cererile de date ale µP fara a adauga starile de asteptare generate de operatia de reînprospatare.
Producatorul tehnologiei EDRAM (Ramtron) descrie 4 avantaje:
- legarea cache-ului SRAM cu DRAM-ul pe acelasi cip presupune folosirea unei magistrale largi de 16384 biti, ce poate determina rata de umplere de aproximativ 60GB/sec. Timpul de umplere este de aproape 7 ori mai mic la EDRAM fata de page-mode DRAM (35 µs fata de 250µs);
- modelul Ramtron foloseste o structura de control ce permite ca memoria sa fie încarcata în timp ce sistemul face transferul în moduil burst din cache, fapt ce reduce timpul de acces;
- scrierea în memoria principala poate fi facuta cu timp de asteptare 0. Prima operatie de scriere necesita 7 µs, iar ciclul de scriere de scriere pentru o pagina normala este de 15 µs.
- proiectarea EDRAM permite accesul la bank-urile de cache separate.

CDRAM (Cached Dram)

Memoria CDRAM (realizata de Mitsubishi) adauga o memorie cache pe fiecare cip, utilizând un model de tip asociat; cipul initial de 4 MB au o memorie cache de 2 MB, folosind câte 2 buffere de 16 biti pentru transferul dintre cache si circuitele externe.

Spre deosebire de EDRAM, partile CDRAM asociaza atât cache–ului cât si DRAM-ului principal aceasi adresa, însa ele pot opera independent una de cealalta. Cache-ul este suficent de rapid pentru a transfera date în modul burst la o frecventa de 100MHz.

Rambus DRAM

Modelul Rambus foloseste un cache de RAM static de 2KB, care se leaga la memoria dinamica printr-o magistrala foarte larga ce permite într-un singur ciclu transferal unei pagini în cache. Cache-ul este destul de rapid, furnizând date la un timp de acces de 15µs.

Rambus opereaza ca o mica retea, trimitând date în pachete cu o lungime de pâna la 256 bytes. Modelul este schimbat radical si cere o modificare a PC-ului pe care opereaza fiind util mai ales pentru sisteme care include integrare video. Colaborarea Intel-Rambus a dus la PC-uri ce utilizeaza RDAM la o 100MHz pe PC-uri de 64 de piti.

MD-RAM (Multibank DRAM)

În locul unui bloc de celule, în care fiecare celula este adunata prin numarul liniei si coloanei, memoria produsa de MoSys Inc. desparte informatia stocata într-un numar de bank-uri de memorie separata.

Modelul MDRAM initial de 4MB, contine 16 bank-uri de câte 256KB, ele fiind legate printr-o magistrala centrala de date ce acceseaza fiecare bank individual.

Acest model permite unui bank de memorie a trimita sau sa primeasca date într-un singur ciclu de ceas, sa comute cu un alt bank apoi pentru un alt transfer.

Deoarece fiecare bank de memorie dispune de o interfata de 32 de biti care lucreaza ca si SDRAM, cipurile MDRAM opereaza la viteze de transfer pâna la un 1GB/s.

VRAM (Video RAM)

Probleme de acces la memorie apar cu precadere în sisteme video, la care memoria este folosita ca un buffer de cadru pentru imaginea de pe ecran, care este înregistrata digital si alocata pentru fiecare element al imaginii. Întregul continut al buffer-ului este citit de 44 ÷80 ori pe secunda. Între timp , PC-ul poate încerca sa scrie o noua informatie în buffer pentru ca aceasta sa apara pe ecran.

Cu memorii DRAM obisnuite, aceste operatii de citire si scriere nu pot apare simultan, una trebuind sa o astepte pe cealalta, timpul de asteptare afectând performantele video.

Asteptarea se evita cu un tip special memorie, care sa aiba 2 cai pentru accesul fiecarui locatii. O astfel de memorie permite citire si scriere simultana; cipurile de memorie video (VRAM) permit citirea completa si scrierea aleatoare la un port, în timp ce la celalalt port se permite doar citirea secventiala, ce corespunde nevoilor de scanare a unei imagini video.

Principalul dezavantaj al memoriei VRAM este costul acesteia. Folosind însa VRAM, se poate mari vsistemului video cu ≈40%

WRAM (Windows RAM)

Un model VRAM cu 2 porturi este WRAM elaborat de Samsung, util la sistemele video proiectate sa asiste o interfata grafica gen Windows .

Cipul de baza WRAM pastreaza 8MB aranjati în plane de 32 biti, fiecare fiind compus din 512 x 512 celule. 4 cipuri asigura memoria necesara pentru a afisa o rezolutie de 1024 x 768 pentru operare în True Color pe 24 biti.

Intern, o magistrala de date de 256 biti leaga fiecare plan de biti cu controlul logic intern al cipului care multiplexeaza datele pe 32 de biti, compatibile cu circuitele PC-uri. Cipul contine 2 registre. Cipul încarca unul din registru, iar din celalalt extrage datele, comutând între ele în momentul în care al doilea se goleste. Aceste registre sunt destinate deci scanarii video. Cipul mai contine înca 4 registre de 32 de biti, doua pentru înmagazinare si culorile de fundal si 2 pentru control si masti.

Cu o rata de transfer de pâna la 640 MB/s, modelul WRAM poate marii viteza sistemului cu pâna la 50% în comparatie cu VRAM.

6.4. Limitarile memoriei interne. Memoria CACHE

Daca memoria nu raspunde întimp la solicitarile microprocesorului, induce stari de asteptare. Timpul de acces este una din limitatrile esentiale ale performantelor calculatoarelor.

Timpul de acces nu este singura variabila de descriere a performantelor cipului de memorie. Mai relevant este ciclul de memorie care masoara cât de repede pot fi efectuate doua accesari consecutive. Ciclul de memorie este, în general, de doua pâna la trei ori mai mare decât timpul de acces. La 25 MHz, de exemplu, un ciclu de ceas este de 40 ns, iar procesorul necesita cel putin doua cicluri între doua operatii cu memoria, deci un total de 80 ns.

Un caz aparte îl constituie SRAM, al caror ciclu egaleaza timpul de acces, operând chiar si mai rapid. Cipurile RAM static sunt disponibile la 25-30 ns, în timp ce DRAM sunt evaluate la 60-70 ns, dar cipurile statice sunt mult mai scumpe, astfel ca rareori sunt folosite pentru memoria de lucru a PC-urilor.

La calculatoarele actuale, cea mai potrivita tehnica este introducerea de memorie cache. O memorie cache interpune un bloc de memorie rapida SRAM între procesor si un bloc de DRAM. Un circuit special, numit controler de cache, încearca sa tina în memoria cache datele sau instructiunile pe care procesorul le va solicita în momentul urmator. Daca informatia ceruta se afla în cache, ea va fi tranferata fara stari de asteptare; daca informatia ceruta nu se afla în cache, ea va fi transferata din RAM cu viteza coespunzatoare RAM-ului, constituind un esec cache.

Un factor major ce determina performantele cache-ului este cantitatea de informatie continuta; cu cât cache-ul este mai mare, cu atât cantitatea de informatie este mai mare. Cel mai bun cache ar fi la fel de mare ca întregul sistem de memorie, dar un astfel de cache este absurd. Uzual, cache-ul este cuprins între 1 MB si câtiva MB. Sistemele de operare multitasking folosesc folosesc cache cu o marime între 256 KB si 2 MB.

Cel mai întâlnit mod de evaluare a performantelor sistemului de memorie la un PC este determinarea ciclurilor de memorie necesare pentru a transfera o linie de memorie, evaluare efectuata printr-o serie 4 numere care reprezinta performanta maxima.

Cea mai performanta memorie este 1-1-1-1, nesitând un ciclu pentru fiecare transfer de cuvânt dublu. Modul burst de tranfer al micoprocesoarelor 486 si Pentium cere 2 cicluri pentru primul transfer (unul pentru stabilirea adresei, celalalt pentru citire sau scriere de date), astfel ca evaluarea practica este 2-1-1-1.

Configuratia logica a memoriei cache se refera la modul în care este aranjata memoria în cache si la modul în care este adresata, ceea ce reprezinta, de fapt, modalitatea în care procesorul stabileste daca informatia ceruta la un moment dat este sau nu disponobila în cache. Principalele optiuni arhitecturale sunt: mapare directa, asociativitate completa si asociativitate pe set.

Memoria cache mapata direct divide memoria cache în blocuri adresate pe linii, corespunzatoare liniilor de încarcare folosite de microprocesoare (Intel pe 32 de biti permite adresarea în multipli de 16 bytes a blocurilor de 128 biti), fiecare linie fiind identificata cu un bit de index. Memoria interna este divizata în blocuri, astfel ca liniile de cache corespund locatiilor blocurilor respective de memorie; poate fi apelata orice linie dintr-un bloc de memorie dar numai din locatiile corespunzatoare din cache. Blocul a carei linie este apelata este identificat cu o eticheta. Controlerul de cache determina daca un byte este stocat într-un cache direct mapat, contolând existenta etichetei specificate prin valoarea bitului de index.
Problema maparii directe este ca un program poate solicita date localizate cu aceleasi indexuri în blocuri de memorie diferite, cache-ul necesitând reîmprospatari continue – ceea ce echivaleaza cu esecuri cache.
Modalitatea opusa de abordare a arhitecturii mapate direct o constituie memoria cache complet asociativa. În acest model, fiecare linie a cache-ului poate fi asociata cu orice bloc al memoriei interne. Controlerul de cache verifica adresele fiecarei linii din memoria cache pentru a determina daca o cerere de memorie a procesorului este o reusita sau un esec. Cu cât sunt mai multe linii de verificat, cu atât timpul este mai mare.

Un compromis între memoria cache mapata direct si cea complet asociativa este memoria cache asociativa pe set, care divide memoria cache în mai multe blocuri mapate direct. Cache-ul este descris prin numarul de moduri în care este divizat. Astfel un cache asociativ pe patru cai este format din patru cache-uri mapate direct. Acest aranjament rezolva problema deplasarii între blocuri cu aceleasi indexuri. Cu cât sunt mai multe moduri pentru un cache, cu atât mai mult va cauta controlerul de cache sa determine daca informatia solicitata se afla sau nu în cache. Acest fapt mareste timpul de acces, diminuând avantajul împartirii pe seturi. Majoritatea producatorilor de PC-uri considera cache-ul asociativ pe patru cai ca fiind optim ca raport performanta/complexitate.

O modalitate de a diminua asteptarea este recurgerea la arhitecturi cache de transfer în modul burst. Ca si în cazul memoriei principale, transferul de tip burst elimina necesitatea trimiterii unei adrese diferite pentru fiecare operatie de scriere/citire a memoriei, orice operatie identificând o secventa de adrese adiacente. În functie de operatia ceruta, un cache în mod burst poate reduce timpul de acces cu pâna la 54%.

Pentru microprocesor, memoria cache poate fi interna sau externa. Un cache intern, numit L1 sau cache primar sau cache de nivel 1, este inclus pe cipul microprocesorului, iar un cache extern, numit cache L2 sau cache secundar, foloseste un controler extern si cipuri de memorie externa.
Cache-ul primar detine un potential de accelarare mai mare decât cel extern din cauza conectarii sale directe la circuitul intern al microprocesorului. La procesoarele Pentium, transferul datelor dintre cache-ul intern si celelalte componente ale sale foloseste o magistrala de 128 de biti, care necesita doua cicluri, datorita magistralei de date de 64 de biti.

Exista cache-uri secundare implementate pe magistrale de 128 de biti, având un mod de adresare orientat pe linie - streamlined (burst mode). Cache-ul intern pastreaza însa un avantaj de doua pâna la cinci ori asupra acestui mod de adresare avansat.

Un cache este folosit pentru stocarea oricarui tip de informatie (instructiuni sau date), fiind denumit cache unificat, sau este împartit functional în cache de instructiuni si cache de date. Aceasta împartire poate duce la îmbunatatirea performantelor, fiind folosita la Pentium, la câteva procesoare Motorola si la majoritatea procesoarelor RISC.

Cache-urile L1 difera si prin modul în care trateaza scrierea în memorie. Unele cache-uri nici nu încearca sa accelereze operatia de scriere, ele transferând comenzile de scriere în cache, scriind apoi în memoria principala cu starile de asteptare corespunzatoare. Aceasta scriere prin cache (write through) este justificata, deoarece garanteaza sincronizarea transferului între memoria principala si cache.

Alternativa mai rapida este write back cache ce permite sa rescrie schimbarile efectuate în cache, controlerul cache actualizând eventual datele schimbate si în memoria principala.

Pentru a adauga flexibilitate si expandabilitate cache-urilor secundare, în calculatoarele ce au proiectate seturile de cipuri corespunzatoare, Intel a lansat propriul standard pentru cache, numit cache-on-a-stick (coast). Modelul de baza foloseste un conector CELP de 160 de pini. Pentru a fi instalat este necesar ca placa de baza sa dispuna de un slot de cache secundar compatibil COAST. Placile cache coast seamana cu cu SIMM-urile normale, diferenta fiind numarul de contacte de la baza placii.

Mai mult decât un dispozitiv, se poate spune despre cache ca este un «principiu». Exista multe feluri de cache, unele constituite din hardware special, iar altele care sunt chiar programme. Cache-ul este un dispozitiv care poate sa lipseasca; lipsa lui se va remarca printr-o viteza mai scazuta, dar nu printr-o reducere a functionalitatii. Un cache este util numai daca anumite informatii sunt folosite frecvent. Dupa un timp cache-ul se umple cu date si pentru a încarca date noi, decizia privind care date vor fi evacuate este foarte importanta din punct de vedere al eficientei. În acest scop se folosesc o serie de algoritmi printre care: Random Policy – aleator, Round Robing – circular, Last Frequenly Used – politica celor mai rar folosite, FIFO – politica primul intrat, primul iesit, algoritmul setului de lucru, algoritmul ceasului, algoritmul celei de-a doua sanse.

O data cu prezenta unui cache, datele devin duplicate în cache. Când se fac scrieri, în functie de circumstante, exista mai multe situatii cu privire la care dintre copii trebuie modificata (din memorie sau din cache).

Similar memoriei cache, orice sistem de operare modern are si un cache de disc. Cache-ul de disc este una din cele mai mari surse de eficienta dintr-un sistem. Aceasta se datoreaza faptului ca diferenta între timpul de acces la disc si cel de acces la mamorie este uriasa. (timpul de aces la disc este de aprox. 10 ms, în timp ce timpul de acces la memorie este de 60 ns sau mai mic). În prezent mai multi producatori de discuri dure sau CD R/W produc discuri cu memorie cache de 2-8 Mb.

6.4.1 Memoria intermediara (cache)

Unul din aspectele cele mai provocatoare ale proiectarii calculatoarelor de-a lungul istoriei a fost realizarea unui sistem de memorie capabil sa fumizeze operanzi procesului la viteza la care îi poate prelucra. Rata înalta recenta de crestere a vitezei procesorului nu a fost însotita de o crestere corespunzatoare a vitezei memoriilor. Fata de procesoare, memoriile s-au dezvoltat mai lent. Data fiind iinportanta enorma a memoriei primare, aceasta situaie a limitat puternic dezvoltarea de sisteme de înalta performanta si a stimulat cercetari în jurul problemei vitezei memoriei care este mult mai mica decât viteza UCP, problema care se înrautateste în fiecare an.

Procesoarele moderne plaseaza cereri covârsitoare la un sistem de memorie, în termeni de latenta (întârzierea în furnizarea operandului) si rata (cantitatea de date furnizate în unitatea de timp). Din nefericire, aceste doua aspecte ale unui sistem de memorie sunt foarte complicate. Multe tehnici pentru cresterea ratei fac acest lucru numai prin cresterea latentei. De exemplu, tehnidile benzii de asamblare utilizate în Mic-3 se pot aplica unui sistem de memorie, cu cereri multiple suprapuse, gestionate eficient. Din nefericire, acestea conduc la o latenta mai mare pentru operatiile individuale de memorie. Pe masura ce frecventa ceasului procesorului devine mai mare, devine din ce în ce mai dificil sa se realizeze un sistem de memorie capabil sa furnizeze operanzi în unul sau doua cicluri de ceas.

O cale de abordare a acestei probleme este sa se prevada memorii intermediare. Asa cum am vazut în sectiunea 6.4, o memorie intermediara pastreaza cuvintele de memorie cel mai recent utilizate într-o memorie mica si rapida, marind viteza de acces la acestea. Daca un procent suficient de mare de cuvinte de memorie necesare se gasesc în memoria intermediara, latenta efectiva de memorie se poate reduce enorm.

Una din tehnicile cele mai eficiente pentru îmbunatatirea ratei si a latentei provine din utilizarea memoriilor intennediare multiple. O tehnica de baza care lucreaza foarte eficient este sa se introduca o memorie intermediara separata pentru instructiuni si pentru date. Exista o serie de avantaje în a avea memorii intermediare separate pentru instructiuni si pentru date, adeseori numindu-se memorie intermediara divizata (split cache). În primul rând, operatiile de memorie se pot initia independent în fiecare memorie intermediara, dublând efectiv rata sistemului de memorie. Acesta este motivul pentru care

trebuie sa existe doua porturi separate de memorie, asa cum am facut la Mic-1: fiecare port are memorie intermediara proprie. De notat ca fiecare memorie intermediara are acces independent la memoria principala.

Astazi multe sisteme de memorie sunt mai complicate decât acesta si o memorie intermediara aditionala, numita memorie intermediara de nivel 2 (level 2 cache), poate exista între memoria intermediara de instructiuni si date si între memoria principala. De fapt, pot exista trei sau mai multe niveluri de memorie intermediara pe masura ce sunt necesare sistem de memorie mai sofisticate. În figura 4-37 vedem un sistem cu trei nivele de memorie intermediara. Cipul UCP însusi contine un memorie intermediara mic de instructiuni si un memorie intermediara mic de date, tipic de 16 KB la 64 KB. Apoi exista nivelul 2 memorie intermediara, care nu este pe cipul UCP, dar poate fi inclus în pachetul UCP, lânga cipul UCP si conectat la acesta printr-o cale de mare viteza. Aceasta memorie intermediara este în general unificata , continând un amestec de instructiuni si date. O dimensiune tipica pentru memoria intermediara de nivel 2 (L2 cache) este cuprinsa 512 KB si 1MB. Memoria interme diara de nivel trei se afla pe placa procesorului si consta în câtiva megaocteti de SRAM, care este mult mai rapid dccât memoria principala DRAM. Memoriile intermediare sunt în general imbricate, având întregul continut al primului nivel inclus în nivelul 2, iar întregul continut al nivelului 2 inclus în nivelul 3.

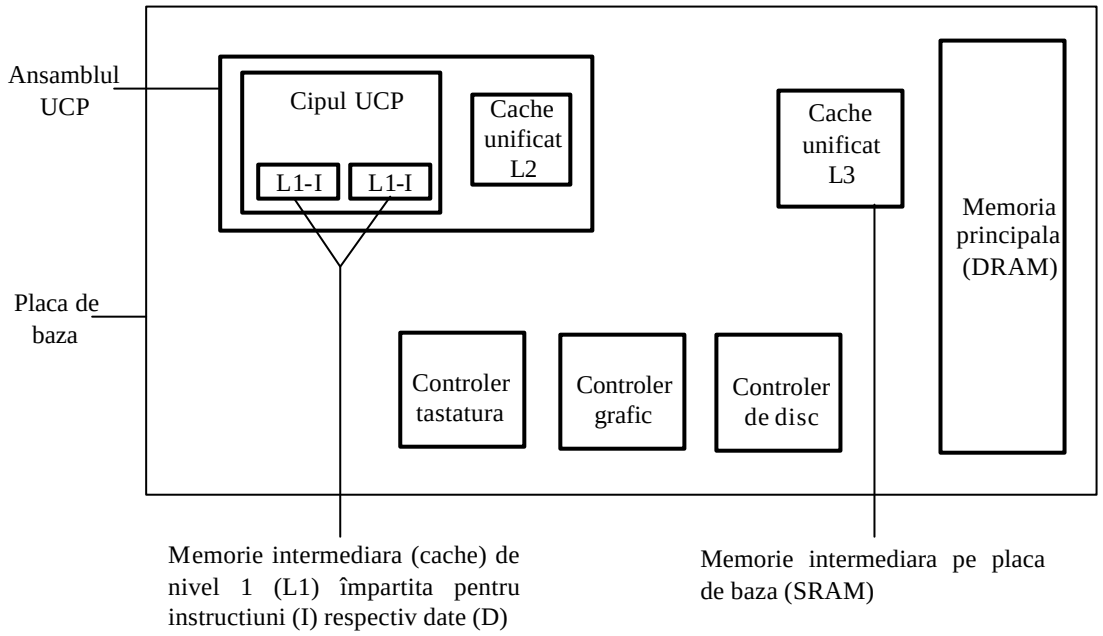

Fig. 6.5. Un sistem cu trei nivele de memorie intermediara .

Pentru a-si atinge obiectivul memoriile intermediare depind de doua tipuri de localitate de adresa. Localitatea spatiala (spatial locality) consta în observatia ca locatiile de memorie cu adrese similare numeric cu o locatie de memorie accesata recent urmeaza probabil sa fie accesate în viitorul apropiat. Memoriile intermediate exploateaza aceasta proprietate aducând mai multe date decât au fost cerute, cu speranta ca viitoarele cereri pot fi anticipate. Localitatea temporala (temporal locality) apare atunci când locatii de memorie accesate recent sunt accesate din nou. Aceasta poate sa apara, de exemplu, la locatiile de memorie din apropierea vârfului stivei sau la instructiunile din interiorul unei bucle. În proiectarea memoriilor intermediare localitatea temporala este exploatata în primul rând prin alegerea a ceea ce se descarca la o ratare în memoria intermediara . Numero algoritmi de înlocuire în memoria intermediara exploateaza localitatea temporala descarcând acele intrari care nu au fost accesate recent.

Toate memoriile intermediare utilizeaza urmatorul model. Memoria principala este împatita în blocuri de dimensiune fixa numite linii de memorie internediara . O linie de memorie intermediara este alcatuita de obicei din 4 pâna la 64 de octeti consecutivi. Liniile sunt numerotate consecutiv începând de la

0, astfel, cu o dimensiune de linie de 32 de octeti, linia 0 este formata din octetii 0 la 31, linia 1 este formata din octetii 32 la 63 si a mai departe. În orice moment anumite linii se gasesc în memoria intenmediara. Când memoria este referita, circuitul de control al memoriei intermediare verifica daca cuvântul referit exista deja în memoria intenmediara. Daca da, valoarea se poate utiliza, fara sa se mai acceseze memoria pnincipala . Daca cuvântul nu este acolo, este eliminata o anumita linie din memoria intermediara , iar linia ceruta este adusa din memorie sau dintr-o memorie intermediara de nivel mai mic pentru a o înlocui. Exista multe variatii ale acestei scheme, dar în toate acestea idea este sa se pastreze cele mai intens utilizate linii din memoria intermediara cât mai mult posibil, pentru a maximiza numarul de referinte de memorie satisfacute de memoria intenmediara.

Memorie intermediara cu corespondenta directa

Cea mai simpla memorie interniediara este memoria intermediara cu orespondenta (direct-mapped cache). Un exemplu de memorie intermediara cu corespondenta directa este aratata în figura 4-38(a). Aceasta memorie intermediara contine 2048 de intrari. Fiecare intrate (rând) din memorie intermediara poate contine exact o linie din memoria principala. Cu o dimensiune a liniei memoriei interniediare de 32 de octeti (pentru acest exemplu), memoria intermediar poate contine 64 de KB. Fiecare mtrare în memoria intermediara consta din trei parti:

1. Bitul Valid indica daca exista vreo data valida în aceasta intrare sau nu. Când sistemul este pornit, toate intrarile sunt marcate ca invalide.

2. Câmpul Etichete (Tag) consta dintr-o valoare de 16 biti unica, identificând linia corespunzatoare din memorie de unde au fost preluate datele.

3. Câmpul Data contine o copie a datelor din memorie. Acest câmp pastreaza o 1inie de memorie intermediara de 32 de octeti.

Într-o memorie intermediara cu corespondenta directa, un cuvânt dat de memorie poa stocat exact într-un singur locc în memoria intermediara . Data fiind o adresa de memorie, exista doar un singur loc în memoria intermediara în care se cauta cuvantul respectiv. Daca acesta nu este acolo atunci nu este în memoria intermediara. Pentru a memora si regasi date în memoria intermediara, adresa este formata din patru componente, ca în Fig. 6.6.(b):

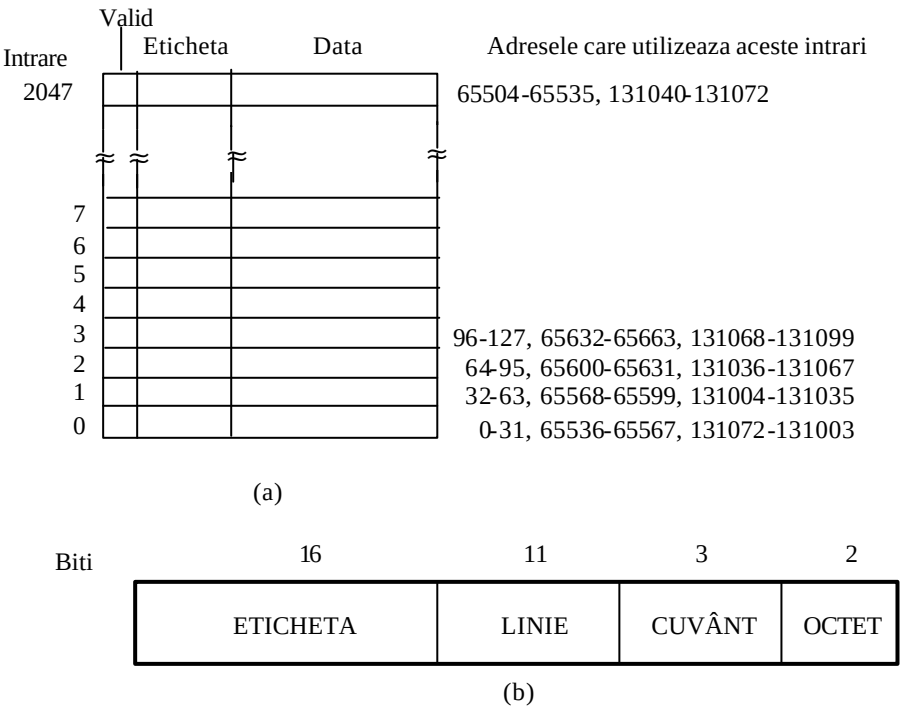

Figura 6.6. (a) Memorie intermediara cu corespondenta directa , (b) O adresa virtuali de 32 de biti.

1. Câmpul TAG corespunde bitilor de etichete memorati în intrarea memoriei intermediare.

2. Câmpul LIME indica intrarea din memoria intenmediara care pastreaza datele corespunzatoare, daca sunt prezente.

3. Campul CUVÂNT indica cuvântul din linie referit.

4. Câmpul OCTET nu este utilizat de obicei, dar, daca este cerut un singur octet, indica octetul din cadrul cuvântului care este cerut. Pentru o memorie intermediara care fumizeaza numai cuvinte de 32 de biti acest cuvânt va fi întotdeauna 0.

Când UCP produce o adresa de memorie, hardware-ul extrage cei 11 biti ai câmpului LINIE din adresa si îi utilizeaza ca index în memoria intermediara pentru a gasi una din cele 2048 de intrari. Daca aceasta intrare este valida, câmpul TAG al adresei de memorie si câmpul Tag al intrarii în memoria intermediara sunt comparate. Daca sunt identice, intrarea în memoria intermediara pastreaza cuvântul cerut, situa numita adresare cu succes (cache hit). În cazul unui succes, un cuvânt care se citeste poate fi luat din memoria memorie intermediara , eliminând necesitatea de a accesa memoria. Doar cuvântul necesar este extras din intrarea în memoria intermediara . Restul intrarii nu este utilizat. Daca intrarea în memoria intermediara este invalida sau etichetele nu se potrivesc, intrarea dorita nu este prezenta în memoria intermediara, situatie numita adresare ratata (cache miss). În acest caz, linia de memorie intermediara de 32 de octeti este citita din memorie si stocata în intrarea în memoria intermediara , inlocuind ceea ce se gasea acolo. Totusi daca dupa încarcare intrarea în memoria intermediara existenta a fost modificata, aceasta trebuie înscrisa înapoi în memoria principala înainte de a fi eliminata.

În ciuda complexitatii deciziei, accesul la un cuvânt necesar poate sa fie remarcabil de rapid. De îndata ce adresa este cunoscuta, locatia exacta a cuvântului este stiuta, daca este prezent în memoria intermediara. Aceasta inseamna ca este posibil sa se citeasca cuvântul din memoria intermediara si sa se livreze procesorului în acelasi timp determinându-se daca este cuvântul corect (prin compararea etichetebor). Astfel procesorul primeste de fapt un cuvânt din memoria intermediara afilând simultan sau poate chiar mai târziu daca este cuvântul cerut.

Aceasta schema de corespondenta pune linii consecutive de memorie în intrari consecutive din memoria intermediara. De fapt, se pot pastra în memoria intermediara pâna la 64 de KB de date adiacente. Totusi, doua linii care difera în adresele lor prin exact 64 de K (65536 de octeti sau orice multiplu întreg al acestui numar, nu se pot memora în memoria intermediara în acelasi timp (pentru ca au aceeasi valoare LINIE) De exemplu, daca un program acceseaza date din locatia X si apoi executa o instructiune, care necesita date din locatia X+65536 (sau orice alta locatie din aceea linie), cea de-a doua instructiune va forta reîncarcarea intrarii în memoria intermediara sa fie reîncarcata , scriind peste ce a fost acolo. Daca aceasta se intâmpla destul de des, poate rezulta o comportare nesatisfacatoare. De fapt, în cazul cel mai defavorabil, folosirea unei memorii intermediare este mai ineficienta decât în cazul în care aceasta ar lipsi, caci fiecare operatie de memorie implica citirea unei întregi linii de memorie intermediara în locul unui singur cuvânt.

Memoriile intermediare cu corespondenta directa reprezinta tipul cel mai comun de memorie intermediara si acestea lucreaza destul de eficient, pentru ca interferentele de genul celei descrise mai înainte pot fi reduse la minim. De exemplu un compilator foarte inteligent poate sa tina seama de interferente când plaseaza instructiuni si date în memorie. Remarcati ca situatia particulara descrisa nu va aparea într-un sistem cu memonii intermediare separate pentru instructiuni si date, deoarece cererile-interferente vor fi servite de memonii intermediare diferite. Astfel putem observa un al doilea beneficiu al folosirii a doua memorii memorie intermediara în loc de una singura: mai multa flexibilitate în tratarea sabloanebor conflictuale de memorie.

Memorii intermediare asociative pe seturi

Asa cum s-a mentionat mai sus, mai multe linii diferite din memorie concureaza pentru aceleasi locuri în memoria intermediara. Daca un program utilizând memoria intermediara din Fig. 4-38(a) utilizeaza frecvent cuvintele de la adresele 0 si 65536, va exista un conflict constant, posibil fiecare refeninta evacuând-o pe cealalta din memoria intermediara. O solutie pentru aceasta problema consta în a admite doua sau mai multe linii în fiecare intrare din memoria intermediara. O memorie intermediara cu n intrari posibile pentru fiecare adresa se numeste memorie intennediara asociativa pe seturi cu n cai (n-way set associative cache). O memorie intermediara asociativa cu patru cai este ilustrata în Fig. 6.6.

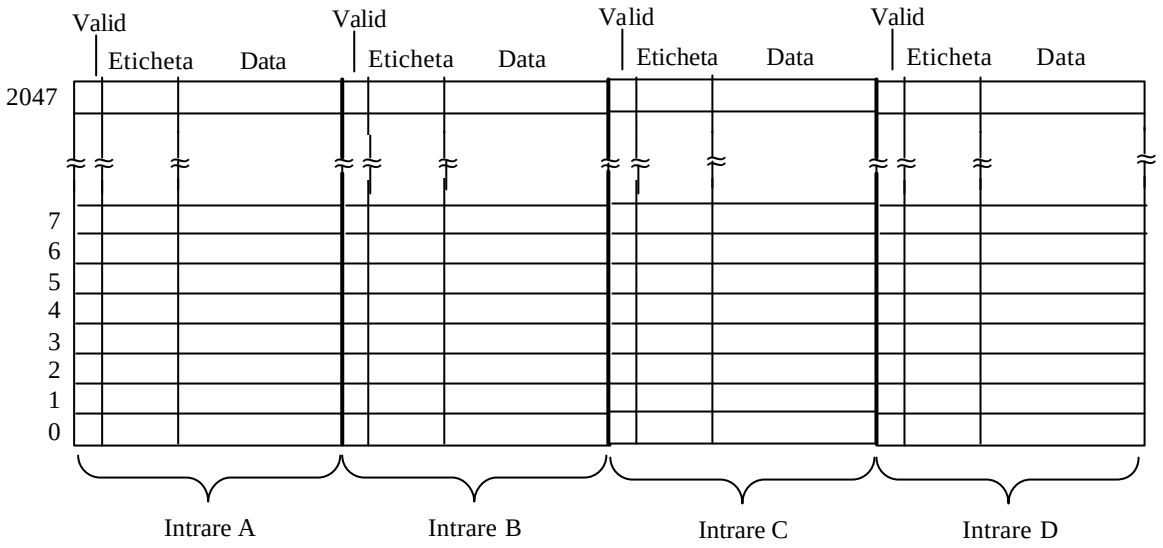

Fig. 6.6. Memorie intermediara asociativa pe seturi cu patru cai

O memorie intermediara asociativa pe seturi este inerent mai complicata decât o memorie intermediara cu corespondenta directa, deoarece, desi intrarea în memoria intermediara ce trebuie examinata se poate calcula din adresa de memorie referita, un set de n intrari în memoria intermediara trebuie verificate pentru a vedea daca linia ceruta este prezenta. Totusi, experienta arata ca memoriile intermediare cu doua cai si patru cai lucreaza suficient de bine încât aceste circuite suplimentare sa-si merite costul.

Utilizarea unei memorii intermediare asociative pe seturi îi lasa proiectantului posibilitatea unei alegeri. Când o noua intrare trebuie adusa în memoria intermediara, care dintre elementele prezente trebuie eliminat? Decizia optima, evident, necesita o privire în viitor, dar un algoritm destul de bun pentru cele mai multe scopuri este LRU (Least Recently Used, rom: cel mai putin recent utilizat). Acest algoritm pastreaza o ordonare a fiecarui set de locatii care pot fi accesate pornind de la o locatie de memorie datâ. Ori de câte ori liniile prezente sunt accesate, se actualizeaza lista, marcând intrarea cea mai recent accesata. Când soseste momentul sa se înlocuiasca o linie, cea de la sfârsitul listei - cea mai puti recent accesata - este cea eliminata.

La limita, putem avea o memorie intermediara cu 2048 de cai continând un singur set de 2048 de itrari de linii. Aici toate adresele de memorie corespund unui singur set, astfel ca accesul necesita compararea adresei cu toate cele 2048 de etichete din memoria intermediara . Observati ca fiecare intrare trebuie sa dispuna acum de o logica de comparare a etichetelor. Deoarece câmpul LINIE are lungimea 0, câmpul TAG ocupa întreaga adresa exceptând câmpurile CUVÂNT si OCTET. Mai mult, când o linie de memorie intennediara este înlocuita toate cele 2048 de locatii sunt candidati posibili pentru înlocuire. Mentinerea unei liste de 2048 de intrari în ordine necesita un efort mare, facând înlocuirea LRU irealizabila. (Sa ne amintim ca aceasta lista trebuie actualizata la fiecare operatie de memorie, nu numai la o ratare). Surprinzator, memoriile intermediare cu asociativitate mare nu îmbunatatesc cu mult performanta fata de memoriile intemiediara cu asociativitate mica, în majoritatea cazurilor, iar în anumite cazuri lucreaza mai prost. Din aceste motive, asociativitatea cu mai mult de patru cai este întâlnita destul de rar.

Înn sfârsit, scrierile pun o problema speciala memoriilor intermediare. Când un procesor scrie un cuvânt si cuvântul este în memoria intermediara, trebuie evident fie sa actualizeze cuvântul, fie sa renunte la intrarea respectiva. Aproape toate modelele actualizeaza memoria intermediara. Dar ce se întâmpla cu actualizarea copiei în memorie? Aceasta operatie trebuie amânata pentru mai târziu, când linia din memorie intermediara este pe punctul de a fi înlocuita de algoritmul LRU. Aceasta alegere este dificila si nici o optiune nu este clar preferabila . Actualizarea imediata a intrarii în memoria principala se numeste actualizare imediata (write through). Aceasta abordare este în general mai simplu de implementat si mai sigura, deoarece memoria este întotdeauna la zi - util, de exemplu, daca se produce o eroare si este necesar sa se refaca starea memoriei. Din nefericire, de obicei necesita un trafic mai intens de scriere la memorie,

asa ca implementari mai sofisticate tind sa utilizeze altemativa cunoscuta ca scriere amânata (write deferred) sau scriere înapoi (write back).

O problema asemanatoare care trebuie abordata în cazul scrierilor este urmatoarea: Ce se întâmpla când se efectueaza o scriere într-o locatie care nu este în memoria intermediara? Este necesar sa se aduca datele în memoria intermediara sau doar sa se faca serierea în memorie? Din nou nici unul din raspunsuri nu este întotdeauna optim. Majoritatea vaniantelor de proiectare care amâna scrierile în memorie tind sa aduca datele în memoria intermediara în urma unei ratari la scriere, aceasta tehnica fiind cunoscuta ca alocare la scriere (write allocation). Pe de alta parte, majoritatea variantelor de proiectare care utilizeaza actualizarea imediata, tind sa nu aloce o intrare la o scriere, deoarece aceasta optiune complica un proiect altfel simplu. Alocarea la scriere este mai buna numai daca exista scrieri repetate în aceleasi cuvinte sau în cuvinte diferite dintr-o linie de memorie intermediara.

4.5.2 Predictia ramificatiilor

Chestiunea a fost tratata în capitolul 5, subcapitolul 5.2.5.